(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 September 2002 (19.09.2002)

PCT

(10) International Publication Number
WO 02/073933 A1

(51) International Patent Classification[7]: H04L 29/12, 29/06

(21) International Application Number: PCT/GB02/00970

(22) International Filing Date: 5 March 2002 (05.03.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
01302109.2 8 March 2001 (08.03.2001) EP

(71) Applicant *(for all designated States except US)*: **BRITISH TELECOMMUNICATIONS PUBLIC LIMITED COMPANY** [GB/GB]; 81 Newgate Street, London, EC1A 7AJ (GB).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **HOVELL, Peter** [GB/GB]; 24 Mill Road, Newbourne, Woodbridge, Suffolk, IP12 4NP (GB). **KING, John, Robert** [GB/GB]; 2 Hertfords Place, Chillesford, Woodbridge, Suffolk IP12 3SD (GB).

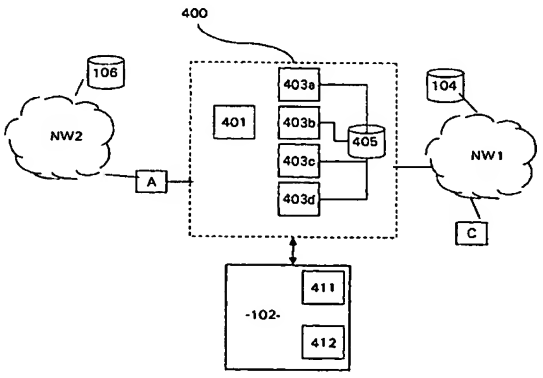
(74) Agent: **ROBINSON, Simon, Benjamin**; BT Group Legal Services, Intellectual Property Department, Holborn Centre, 8th floor, 120 Holborn, London EC1N 2TE (GB).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: ADDRESS TRANSLATOR



(57) Abstract: The present invention concerns network address translation, and provides apparatus for providing communication between a network device in a first network and a network device in a second network, where the first network operates in accordance with a first communication protocol and the second network operates in accordance with a second communication protocol. The apparatus comprises (i) first means for assigning an alias to a target network device in the first network, the alias being compatible with the communication protocol of the second network ; (ii) second means for translating said assigned alias to an address for the target network device, said translated address being compatible with the communication protocol of the first network, wherein the first means and the second means are separately addressable in one or both of said networks, and said assigned alias corresponds to an address of the second means, such that, when a network device in the second network sends one or more communication(s) using an address comprising the assigned alias, the or each communication is routed to the second means, whereupon the second means translates the alias into the address of the target network device in the first network and sends the communication(s) into the first network.

WO 02/073933 A1

WO 02/073933 A1

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

ADDRESS TRANSLATOR

This invention relates to an address translator, and is suitable particularly, but not exclusively, for address translation between different networks.

Currently all commercial Internet Protocol (IP) networks are IPv4 networks; however, at some point in the future, commercial IP networks will be IPv6 networks. In the meantime there will be a transitory period, during which commercial IP networks will comprise a mixture of IPv4 and IPv6 networks. IPv6 is a totally different protocol to IPv4 and is fundamentally incompatible with IPv4. Therefore, during the transitory period at least, network devices and/or networks will require mechanisms to enable a node and/or host in an IPv4 network, having an IPv4 address, to communicate with a node and/or host in an IPv6 network, having an IPv6 address.

Several migration mechanisms have been developed; see for example a document published in November 2000 by the Internet Engineering Task Force (IETF) and available from the IETF at http://www.ietf.org/internet-drafts/draft-ietf-ngtrans-introduction-to-ipv6-transition-05.txt, entitled "An Overview of the Introduction of IPv6 in the Internet", authors: W. Biemolt et al, IETF Status: Draft working towards Informational RFC. Essentially these methods can be categorized as either "aggressive, short term" methods or "conservative, long term" methods.

A problem with at least some of known migration methods is that they have been designed and operated in a test environment, and have not been subjected to the volume of traffic experienced in commercial IP networks. There has therefore been little work carried out on designing migration methods that are commercially scalable and robust. This could be a serious problem, given that the transition between IPv4 and IPv6 is expected to be long, and the volume of IP traffic is continually increasing.

In the following description, the terms "host", "network device", "pool of addresses" and "node" are used and are defined as follows:

"node": any equipment that is attached to a network, including routers, switches, repeaters, hubs, clients, servers; the terms "node", "device" and "network device" are used interchangeably;

"host": equipment for processing applications, which equipment could be either server or client, and may also include a firewall machine; and

"pool of addresses": a group of addresses available for a purpose; the addresses could include IPv4 addresses that are globally unique, or addresses that are private within a network, e.g. a VLAN.

According to a first aspect of the invention there is provided apparatus for providing communication between a network device in a first network and a network device in a second network, where the first network operates in accordance with a first communication protocol and the second network operates in accordance with a second communication protocol. The apparatus comprises

(i) first means for assigning an alias to a target network device in the first network, the alias being compatible with the communication protocol of the second network;

(ii) second means for translating said assigned alias to an address for the target network device, said translated address being compatible with the communication protocol of the first network,

wherein the first means and the second means are separately addressable in one or both of said networks, and said assigned alias corresponds to an address of the second means, such that, when a network device in the second network sends one or more communication(s) using an address comprising the assigned alias, the or each communication is routed to the second means, whereupon the second means translates the alias into the address of the target network device in the first network and sends the communication(s) into the first network.

Particularly advantageous embodiments of the invention are applied between IPv4 and IPv6 networks, so that the first network is a IPv4 network and the second network is a IPv6 network.

Preferably the alias comprises a network address, and when the communication is being sent into an IPv6 network, the network address includes an identifier representative of the second means.

Conveniently the second means comprises a plurality of further devices. Thus, upon assignment of alias to the target network device, the first means effectively causes subsequent communications to occur via one of a plurality of

further devices. Having a plurality of devices advantageously introduces resilience, scalability and efficient management of network loading.

Advantageously the or each further device has access to one or more groups of aliases, and each group can be stored in a store. Alternatively, two or more groups can be stored in a store.

Preferably embodiments include selecting means for selecting one of the plurality of further devices in accordance with predetermined criteria, such as device characteristics. Advantageously the selecting means is operable to monitor the device characteristics, so that selection of a device is based on current device performance. Monitored device characteristics include at least one of operational status of device, loading on device, and/or aliases available to the device.

In preferred embodiments, the selecting means is in operative association with the first means, so that the first means is operable to retrieve an alias available to the further device, which retrieved alias is the assigned alias.

Conveniently embodiments include a mapping store for storing mappings between the assigned alias and the network device assigned to the alias. The mapping store can be managed by the first means, a database, or by the further device. The selection of manager of the mapping stored is typically subject to criteria such as network traffic, ownership of network devices and transmission paths.

According to a second aspect of the present invention there is provided a method of providing communication between a network device in a first network and a network device in a second network corresponding to the apparatus described above.

Further aspects, features and advantages of the present invention will be apparent from the following description of preferred embodiments of the invention, which refer to the accompanying drawings, in which

Figure 1 is a schematic diagram showing a first and a second network and known means facilitating communicating between the first and second networks;

Figure 2 is a flow diagram showing steps carried out by a known address translation process when setting up communications between hosts located in the first and second networks of Figure 1;

Figure 3 is a flow diagram showing steps carried out by a known address translation process when data is sent from a host located in the first network to a host located in the second network of Figure 1;

Figure 4 is a schematic diagram showing an address translator according to an embodiment of the present invention;

Figure 5a is a schematic diagram showing a configuration of address pool comprising part of the address translator shown in Figure 4,

Figure 5b is a schematic diagram showing an alternative configuration of address pool comprising part of the address translator shown in Figure 4,

Figure 5c is a schematic diagram showing yet another possible configuration of address pool comprising part of the address translator shown in Figure 4,

Figure 6 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when setting up communications instigated by a host in the *first* network;

Figure 7 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when data is sent from the instigating host of Figure 6 to a host in the *second* network;

Figure 8 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when data is returned from the host in the *second* network;

Figure 9 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when setting up communications instigated by a host in the *second* network;

Figure 10 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when data is sent from the instigating host of Figure 9 to a host in the *first* network; and

Figure 11 is a flow diagram showing steps carried out by an embodiment of the address translator shown in Figure 4, when data is returned from the host in the *first* network.

Embodiments of the invention are concerned with issues relating to migration from IPv4 to IPv6 networks. Specifically, embodiments of the invention are concerned with scalability aspects of migration methods; as stated above, almost all of the IPv6 networks currently in operation are "test" networks and are not subject

to the volume of IP traffic passing through commercial IP networks. Thus the performance of the migration methods in a commercial environment may be unacceptably low.

One embodiment of the invention is concerned with the Network Address Translator – Protocol Translator (NAT-PT) method, which is documented by the IETF in Request for Comments" RFC2766, available from the IETF at http://www.ietf.org/rfc/rfc2766.txt. NAT-PT is a mechanism that translates both the IP header and the IP addresses from IPv6 to IPv4, and vice versa. With NAT-PT explicit mappings are maintained between arbitrary IPv4 and IPv6 addresses, so that, when converting addresses, NAT-PT consults a pre-configured table to determine the corresponding address to use with the other protocol. As documented in the Introduction section of RFC2766, with NAT-PT, packets that are part of a session between an IPv4 host and an IPv6 host MUST go via the same NAT-PT entity, because address mappings are kept within that NAT-PT and are not shared. This is a consequence of the translation mechanism of NAT-PT: NAT-PT is a stateful translation process, meaning that there is specific information that must be retained in order for each individual session to be translated.

Thus address translation is performed by aliasing an IPv6 address with an IPv4 address in much the same way as is done with a conventional Network Address Translation (NAT) device. Some NAT-PT implementations include a DNS Application Level Gateway (DNS_ALG), which translates DNS requests and responses.

Figure 1 shows a conventional implementation of NAT-PT in operation between a first network NW1, which may be an IPv4 network and a second network NW2, which may be an IPv6 network. The implementation includes a single translator 101, which comprises processes 102 that manage assignment of IPv6/v4 addresses. Such processes 102 include the DNS application level gateway. In addition the translator 101 has access to a pool 103 of IPv4 addresses, which is used for assignment to IPv6 nodes. Figure 1 additionally shows two DNS servers 104, 106, a first 104 of which stores IPv4 name to address mappings in the form of "A" records, and a second 106 of which stores IPv6 name to address mappings in the form of "AAAA" records.

The translator 101 is typically located on a border router, referred to as an ingress interface with respect to the IPv6 network NW2. In general

Conventional operation of this known NAT-PT implementation is shown in Figure 2. In a typical scenario, host C in the IPv4 network sends 202 a name lookup request for host A in the IPv6 network. This lookup request is termed an "A" type DNS request. This request is received 204 by the translator 101, which tags 206 the request as an IPv6 record request, so that the request becomes an "AAAA" type DNS request, and forwards 208 the tagged request to the DNS server 106.

The DNS server 106 replies 210, returning an IPv6 network address to the translator 101, which, in co-operation with the processes 102, assigns 212 an IPv4 address from the pool of addresses to the returned IPv6 address. The translator 101 stores 214 the mapping between the assigned IPv4 address and the returned IPv6 address, and forwards 216 the assigned IPv4 address to the requesting host C.

In subsequent communications between hosts C and A, and as shown in Figure 3, host C sets 302 the destination address to be the assigned IPv4 address for outgoing packets, and sends 304 the packets to the assigned IPv4 address of Host A, and conventional IP routing ensures that the packet routes to the translator 101. The translator 101 then looks up 306 the mapping between assigned IPv4 address and IPv6 address to retrieve the IPv6 address of host A, and makes this 308 the destination address of the packet.

For the packets to be routed from the translator 101 to host A, the translator 101 has to modify the source address of the packet, which is the IPv4 address of node C, into IPv6 format. This involves expanding 310 the IPv4 address of host C with a prefix that is representative of the translator 101. As is well known, an IPv4 address is 32bits long, whereas an IPv6 address is 128bits long. As stated above, an IPv6 host cannot interpret an IPv4 address, and vice-versa – because of the differences in address length. Thus when an IPv4 packet arrives at the translator 101 a 96bit prefix, which is indicative of the translator 101, is added to the source address of the packet (32bits) to make an IPv6 address (128bits). Packets sent to this IPv6 address will then be routed to the translator 101. [For example an IPv4 source address 10.10.10.10 arriving at the translator 101 could be given the prefix 2001:618:1:2:: so that the source IPv4 host has the following address in the IPv6 world: 2001:618:1:2::10.10.10.10. An IPv6 packet sent to this address would go to translator 101 because the prefix 2001:628:1:2:: routes to the translator 101.]

The translator 101 then sends 312 the packet, using the expanded IPv6 address. All subsequent communications between host A and host C can make use of the mappings stored in the translator 101.

Communications initiated by host A, in the IPv6 network, involve similar address assignment; for a working example the reader is referred to the RFC detailed above.

From the above it can be seen that, once IPv6 addresses have been assigned, the translator 101 performs translation of packets as they pass between hosts (C) in the IPv4 network and hosts (A) in the IPv6 network, thus acting as a medium for all communication between said hosts A, C. A problem with this configuration is that centralized address assignment and communications processing could present scalability problems when IPv6 networks become mainstream.

The essence of embodiments of the invention is that the functionality of initial address assignment (Figure 2) is separated from subsequent events that make use of the assigned addresses (e.g. packet encapsulation as packets pass from host A to host C, as shown in Figure 3). This is a particularly unexpected development of the NAT-PT method described in RFC2766: the RFC mandates that, because NAT-PT is stateful, address assignment and allocation should be carried out on one and the same device – i.e. that they are inseparable.

In one embodiment of the invention, and as shown in Figure 4, the translator 400 essentially comprises two components: a controller 401 and a plurality of devices 403a, 403b ... 403n (hereinafter collectively referred to as 403_i). Requests to communicate with hosts in the IPv"other" network are received by the controller 401, which polls the devices 403_i in order to identify one that has capacity to service the request. A device so identified thereafter deals with all subsequent communication between hosts in IPv6 and IPv4, and the subsequent communication is therefore independent of the controller operations. The configuration and functionality of these components is described in more detail below.

Thus in embodiments of the invention, address assignment events are separated from subsequent communication events between hosts in IPv4 and IPv6 networks (e.g. translating packets using the assigned addresses). In addition, the controller 401 can select from a plurality of devices for address assignment.

There are several advantages associated with the embodiments:

- Firstly devices are passive in the address assignment phase (Figure 2), which means that if a device fails it won't affect servicing of new requests (the controller 401 can select another device, for example).
- Secondly the translator 400 is scalable, as devices can easily be added in accordance with demand.
- Thirdly, communication between devices in IPv4 and IPv6 networks can continue independently of address assignment, so that (i) failure of controller 401 due to address assignment problems does not affect communications that have already been established between hosts in IPv4 and IPv6 networks; (ii) network problems corresponding to communications between IPv4 and IPv6 hosts do not affect the controller; and (iii) the controller does not have to account for, or schedule, subsequent communication events.
- Fourthly, as IPv6 begins to outgrow IPv4 then devices can be de-commissioned as required, again without major translation service disruption.
- Fifthly, for an IP network provider to deploy an IPv6 network with "IPv6-only" services, it will be necessary to provide some kind of translation facility so that both IPv4 network users can access the IPv6 services and IPv6 users can access existing IPv4 based services. The embodiments presented herein allow a commercial service provider the ability to dynamically scale the address translation service according to the dimensions of the IPv6 network.

Referring back to Figure 4, the configuration of the translator 400 will now be described in more detail.

The controller 401 receives DNS requests initiated by hosts in either the IPv4 or the IPv6 networks, and manages DNS lookups, in the manner described above, in respect of the requests. Having received a returned IPvn address from a respective DNS server 104, 106, the controller 401 then identifies a device 403_i that will mediate for subsequent communications between the requesting host (in the example above, host C) and the destination host (in the example above, host A). Each device has a globally routable prefix (i.e. a prefix that is appended to destination addresses of packets destined for the device), which, when appended to an IPv4 address, enables packets to reach the device (as described above).

In one embodiment identification of a device 403_i comprises determining whether the device 403_i is up and running. In addition the controller 401 identifies

whether there are any free addresses available to the device 403_i, and the current loading on, or the number of communications that are currently being handled by, a device. Conveniently the controller 401 may run a program 411, which polls each device 403_i at predetermined intervals to determine current loading, operational status and IP addresses accessible to that device. The program 411 gathers the loading and address availability data from the devices 403_i, and stores it, for example as a list in memory. The controller 401 may run the program 411 at predetermined intervals, for example every second, or as frequently as required to capture changes to the devices 403_i.

The controller 401 may also run selection algorithms for selecting a device 403_a from the list. For example, a typical selection algorithm 412 searches the list for an operational device that has access to at least one free IPv4 address and that has a loading below a predetermined threshold. If more than one device satisfies these criteria, then the device with lowest loading is selected. Many variations on this example are possible, and would be apparent to the skilled person.

In one embodiment each device 403_i may be a conventional router, so that the controller 401 can derive the loading on a device 403_a by issuing Simple Network Management Protocol (SNMP) messages to a Management Information Base (MIB) that is maintained on the router. SNMP is part of the known TCP/IP network software, and MIB, or Management Information Base, is a standard specifying the data items that a host, router or switch must keep, together with the operations allowed on each. SNMP is the protocol that enables information to be extracted from a MIB, and is known to those skilled in the art. For further details see Request for Comments (RFC) 2037/2737, Entity MIB, McCloghnie *et al* 1996/1999, published by the Internet Engineering Task Force (IETF) (available from http://www.ietf.org), or Understanding SNMP MIBs by David Perkins, Evan McGinnis. Prentice Hall, 1st edition (December 3, 1996).

In one of the embodiments each of the devices 403_i has access to a pool 405 of IPv4 addresses, and the availability, to any single device 403_a of IP addresses, is recorded on a respective device 403. The controller 405 could therefore determine address availability per device 403 by reviewing the record of address availability thereon. The skilled person would realize that such a record does not need to be stored on the devices themselves, but could be held centrally, e.g. in a database.

The pool 405 can either be a central pool, as is shown in Figure 4, and accessible by all of the devices 403_i, or can comprise a plurality of pools, each accessible to a limited number of the devices 403_i. For example, there may be one pool 405 per device 403_i, as shown in Figure 5a, or there may be one pool 405 per n devices 403_i, as shown in Figures 5b and 5c.

Essentially the controller 401 may comprise one or more programs running on a processor, such as a conventional client or server computer. Alternatively the controller 401 could run on a router. In either configuration, the controller 401 could connect to each of the devices 403_i via dedicated links or via the Internet; if security were a consideration, dedicated links would be more suitable. As an alternative, a security protocol, such as IPsec, which is a mandatory part of IPv6, could be used for communications between controller 401 and devices 403_i. The programs can include socket processes that listen for incoming DNS lookup requests and listen for requests from devices 403_i.

In preferred embodiments the controller 401 processes incoming requests on a FIFO (First-in, First-out) basis – i.e. the controller 401 implements a queuing discipline in which entities (here requests, incoming packets) are stored in a queue (or in the stack) and are serviced in the same order in which they arrive. As an alternative, the controller 401 could process requests in accordance with LIFO (last-in, first-out), where the most recent request is handled next and the oldest request doesn't get handled until it is the only remaining request on the queue (or in the stack).

Other scheduling policies are possible, e.g. when the controller 401 is subject to constraints; a suitable scheduling policy could utilise some sort of heuristic method (or combination of heuristic methods) in an attempt to schedule the requests so as to satisfy the constraint criteria. As a further alternative, the scheduling policy could make use of Quality of Service information included in IPv6 headers: certain bits in the IP header indicate the priority of the request, and the controller 401 could include means for examining these bits (not shown). The controller 401 could have a plurality of queues, each corresponding to a different priority level, which the controller 401 services on, e.g. a sequential basis.

A flow chart for the translator 400 managing communications initiated by a host C in an IPv4 network (to communicate with a host A) is shown in Figures 6 - 8,

and a flow chart for the translator managing communications initiated by a host A in an IPv6 network (to communicate with a host C) is shown in Figures 9 - 11.

Figures 6 and 9 each show a main loop, on the left hand side of the respective flow charts, representing processes carried out in respect of incoming initiating requests. On the right hand side of the respective flow charts there is a sub-loop, which represents processes carried out between the controller 401 and devices 403_i.

Considering firstly the case shown in Figure 6, host C in an IPv4 network sends 602 a name lookup request for host A in the IPv6 network. This request is received 604 by the controller 401, which tags 606 the request as an IPv6 record request, by modifying the request to a "AAAA" record request, and forwards 608 the tagged request to the DNS server 106. The DNS server 106 replies 610, returning an IPv6 network address to the controller 401.

The controller 401 then tries to identify a device 403_a for mediating communications. This process is shown in the loop on the right hand side of Figure 6 and comprises the following steps: the controller 401 accesses 612 a list of devices 403_i that, for each device 403_i, details loading on the device and IP addresses accessible to that device. Such a list can be stored in memory, as described above. The controller 401 then applies 614 a selection algorithm 412, such as the one described above, to the list in order to identify a device for the current request. Once a device 403_a has been identified 616, an IPv4 address available to that device 403_a is selected and returned 618 to the controller 401. (In addition to, or in place of, the automatic polling of devices 403_i described above, the process 411 may automatically be invoked each time a device is so identified).

The controller assigns 620 the IPv4 address returned at step 618 to the IPv6 address returned at step 610 and saves a mapping between the two. The assigned IPv4 address is then sent 622 to host C. Typically, a number of IP addresses that are available to a device will be pre-assigned to an interface of that device (in a manner known to those skilled in the art), so that a single interface effectively has a plurality of IP addresses. Thus when a packet is sent from host C bearing a destination address of the assigned IPv4 address, the packet will be routed to the corresponding interface of the device 403_a.

At this point host C has an IPv4 address for host A, which it can use to route packets to host A. Figure 7 shows the scenario of host C sending 702 a packet to host A: the packet has a destination address set to the assigned IPv4 address and source address set to the IPv4 address of host C. As the assigned address is routable to the identified device 403_a (as described in the paragraph above) the packet will arrive at the device 403_a, whereupon the device 403_a sends 704 a request to the controller 401 for the IPv6 address corresponding to the destination address of the packet (which is the IPv4 address assigned at step 620). The controller 401 performs 706 a lookup of its stored mappings and transmits 708 the returned IPv6 address to the device 403_a. At this point the device 403_a is in receipt of all of the information required to enable it to autonomously mediate further communications between hosts A and host C.

The device 403_a then modifies 710 the source and destination addresses of the packet sent by host C, expanding the source address to include the IPv6 prefix of the device 403_a together with the IPv4 address of the host C (as described above), and setting the destination address to the returned IPv6 address. The device 403_a then sends 712 the packet into the IPv6 network for receipt by host A.

Referring to Figure 8, if host A sends a response to host C, host A sends 802 one or more packets having as destination address the IPv6 prefix of the device 403_a together with the IPv4 address of the host C and as source address its own IPv6 address (which is of course the IPv6 address returned at step 610). The device 403_a receives 804 the, or each packet, and queries 806 the controller 401 for an IPv4 address corresponding to the IPv6 source address of the incoming packet.

The controller 401 returns 808 the corresponding IPv4 address, which is the assigned IPv4 address (from step 620), to the device 403_a, whereupon the source address of the or each incoming packet is replaced 810 with the assigned IPv4 address. In addition the device 403_a modifies 812 the destination address, removing the IPv6 prefix of the device 403_a to leave the IPv4 address of host C. Finally the device 403_a sends 814 the, or each packet, into the IPv4 network.

Considering secondly the case shown in Figure 9, host A in an IPv6 network sends 902 a name lookup request for host C in the IPv4 network. This request is received 904 by the controller 401, which tags 906 the request as an IPv4 record request, by modifying the request to a "A" record request, and forwards 908 the

tagged request to the IPv4 DNS server 104. The DNS server 104 replies 910, returning an IPv4 network address to the controller 401.

The controller 401 then tries to identify a device 403_b for mediating communications. This process is shown in the loop on the right hand side of Figure 9 and comprises the following steps: the controller 401 accesses 912 the list of devices described above, and applies 914 a selection algorithm 412 to the devices on the list in order to identify a device for the current request. Once a device 403_b has been identified 916, the controller 401 reserves 918 an IPv4 address from the pool 405 accessible to the identified device 403_b. The controller 401 then saves 920 a mapping between the reserved IPv4 address and the IPv6 address of host A. Finally the controller 401 appends 922 a prefix identifier of the identified device 403_b (as described above) to the IPv4 address returned at step 910, and sends 924 this to host A.

At this point the device 403_b is in receipt of all of the information required to enable it to autonomously mediate further communications between hosts A and host C.

Figure 10 shows the scenario of host A sending 1002 a packet to host C: the packet has a destination address set to the IPv6 address sent at step 924. As this address is globally routable to the identified device 403_b (as described above) the packet will arrive at the device 403_b, whereupon the device 403_b removes 1004 the prefix from the destination address, leaving the IPv4 address of host C.

The device 403_b sends 1006 a request for the IPv4 address reserved at step 922, whereupon the controller 401 sends 1008 the reserved address. The device 403_b modifies 1010 the source and destination addresses of the packet sent by host A, setting the source address to the IPv4 address of the host C, and setting the destination address to the reserved IPv4 address. The device 403_b then sends 1012 the packet into the IPv4 network for receipt by host C.

Referring to Figure 11, if host C sends 1102 a response to host A, such a response may comprise one or more packets having as destination address the reserved IPv4 address and as source address its own IPv4 address. The device 403_b receives 1104 the, or each packet, and queries 1106 the controller 401 for an IPv6 address corresponding to the destination address of the packet. The controller 401

performs a lookup of its stored mappings and returns 1108 an IPv6 address, which is the IPv6 address of host A.

The device 403_b adds 1110 its prefix to the source address of the packet (IPv4 address of host C) to form an IPv6 address, as described above, and the destination address of the packet is replaced 1112 with the IPv6 address returned at step 1108 (i.e. the IPv6 address of originating host A). Finally the device 403_b sends 1114 the packet into the IPv6 network.

Second embodiment:

The second embodiment includes all of the features described in respect of the first embodiment, but instead of the address mappings being stored on the controller 401, the mappings are stored on a remote database, or similar, that is accessible to both the controller 401 and the devices 403_i. Thus in this embodiment the devices 403_i do not have to communicate with the controller 401 at all once the initial address assignment has been established.

Third embodiment:

The third embodiment includes all of the features described in respect of the first embodiment, but the devices 403_a cache addresses in memory for a predetermined period. This embodiment would be particularly suitable for the scenario shown in Figures 6 - 11 , where a host in an IPv4 network initiates communication, and where an intense period of communication is in progress between hosts A and C. If the device 403_a caches mapping information locally, the device 403_a does not have to continually request address mapping information from the controller. This therefore keeps communication between A and C wholly independent of the controller 401, minimizes network traffic and allows communications dependent on address translation to progress faster than in the other embodiments.

Fourth embodiment:

The fourth embodiment includes all of the features described in respect of the first embodiment, but the mappings are stored in the address pool 405, rather than in the controller 401. As for the second and third embodiments, the devices

403$_i$ do not have to communicate with the controller 401 at all once the initial address assignment has been established.

Fifth embodiment

The fifth embodiment includes all of the features described in respect of the first embodiment, but the allocation of IPv4 addresses from the address pool 405 is managed by the controller 401, rather than by the devices 403$_i$. In such a situation the first and second selection algorithms do not include reviewing IP address availability when identifying a device 403$_i$. In this embodiment the address pool 405 could be stored on a Dynamic Host Configuration Protocol (DHCP) server, so that the controller 401 requests IPv4 addresses in accordance with DHCP. In this situation, (where allocation of IPv4 addresses is managed by the controller 401) then allocated addresses must be configured into the IPv4 interface of the identified device 403$_i$. i.e. an address is chosen from the address pool 405, which is then given it to the identified device 403$_i$.

This embodiment could be used in conjunction with either of the second, third or fourth embodiments.

Sixth embodiment

The sixth embodiment can be used in conjunction with any of the above embodiments. This embodiment is concerned with resilience issues relating to the controller 401: in the event that the controller 401 is operationally inactive or the loading on the controller becomes unacceptably high, some kind of "back-up" system is required.

The sixth embodiment provides a second, or a mirror, controller, which monitors the operational status and loading on the controller 401 in accordance with predetermined criteria. In the event that the mirror controller detects that the controller 401 fails to satisfy one or more of the criteria, it either switches all control over to the mirror controller, which thereafter services the requests in the manner described above, or it balances control between the mirror controller and the controller 401. Preferably an alert is sent to the party operating the translator 400.

A cascaded arrangement of controllers 401$_i$ could be a preferred arrangement, in view of the fact that the number of requests is expected to scale

with the introduction of IPv6 networks. As the controller 401 allocates sessions to the devices 403$_i$ dynamically, the translator 400 could include a plurality of active controllers without requiring any significant changes to the above-described embodiments. Furthermore, the DNS servers 104, 106 could be configured to forward requests to a next available controller in the event that a given controller fails (One of the features of a DNS server is that it can be configured to forward requests for certain domain names to more than one other DNS server. This feature could be employed in connection with the controllers, to achieve the above-described effect).

As disclosed in the document published by the IETF, referred to above (available at http://www.ietf.org/internet-drafts/draft-ietf-ngtrans-introduction-to-ipv6-transition-05.txt), other migration methods include:

- Automatic Tunnels
- Configured Tunnels
- Tunnel Broker
- 6over4
- 6to4
- Dual stack transition mechanism (DSTM)
- Stateless IP/ICMP Translation Algorithm (SIIT)
- Bump-In-the-Stack" Technique (BIS)
- SOCKS64

Some of these methods are naturally scalable – such as SIIT, because it is a stateless mechanism and BIS, because address translation takes place in a host. However, other methods, such as DSTM, suffer from scalability problems similar to those identified for NAT-PT. Embodiments of the invention could thus be integrated with features of the DSTM architecture in order to improve their scalability.

With DSTM, and as is known to those skilled in the art, a dual stack host tunnels IPv4 packets over an IPv6 network to a DSTM border router at the IPv4/IPv6 boundary, where the packets are subsequently un-encapsulated into IPv4 packets. The dual stack host is dynamically assigned an IPv4 address by a DHCP server (to be used as source address for any packets sent into the IPv4 network). In addition, the DHCP server tells the dual stack host of the IPv6 address of the border router (termed "tunnel endpoint"). Embodiments of the present invention could be integrated with the DSTM method so that the DHCP server assigns an IPv6 tunnel endpoint address (border router) according to the loading etc. on available border routers.

Thus, in terms of the elements of the embodiments presented above, the controller 401 would co-operate with the DHCP server and the devices 403_i would be DSTM border routers.

CLAIMS

1. Apparatus for providing communication between a network device in a first network and a network device in a second network, the first network operating in accordance with a first communication protocol and the second network operating in accordance with a second communication protocol, the apparatus comprising

(i) first means for assigning an alias to a target network device in the first network, the alias being compatible with the communication protocol of the second network;

(ii) second means for translating said assigned alias to an address for the target network device, said translated address being compatible with the communication protocol of the first network,

wherein the first means and the second means are separately addressable in one or both of said networks, and said assigned alias corresponds to an address of the second means, such that, when a network device in the second network sends one or more communication(s) using an address comprising the assigned alias, the or each communication is routed to the second means , whereupon the second means translates the alias into the address of the target network device in the first network and sends the communication(s) into the first network.

2. Apparatus according to claim 1, wherein the alias comprises a network address.

3. Apparatus according to claim 2, wherein the network address includes an identifier representative of the second means.

4. Apparatus according to claim 3, wherein the second means comprises a plurality of further devices.

5. Apparatus according to claim 4, wherein the or each further device has access to one or more groups of aliases.

6. Apparatus according to claim 5, wherein each group is stored in a store.

7. Apparatus according to claim 5, wherein two or more groups are stored in a store.

8. Apparatus according to any one of claims 4 to 7, including selecting means for selecting one of the plurality of further devices in accordance with predetermined criteria.

9. Apparatus according to claim 8, wherein the predetermined criteria includes device characteristics.

10. Apparatus according to claim 9, wherein the selecting means is operable to monitor device characteristics.

11. Apparatus according to claim 9 or claim 10, wherein the device characteristics include at least one of operational status of device, loading on device, and/or aliases available to the device.

12. Apparatus according to any one of claims 8 to 11, wherein the selecting means is in operative association with the first means.

13. Apparatus according to any one of claims 4 to 12, wherein the first means is operable to retrieve an alias available to the further device, which retrieved alias is the assigned alias.

14. Apparatus according to any one of claims 4 to 13, including mapping a store for storing mappings between the assigned alias and the target network device assigned to the alias.

15. Apparatus according to claim 14, wherein the mapping store is managed by the first means.

16. Apparatus according to claim 14, wherein the mapping store is managed by a database.

17. Apparatus according to claim 14, wherein the mapping store is managed by the further device.

18. A method of providing communication between a network device in a first network and a network device in a second network, the first network operating in accordance with a first communication protocol and the second network operating in accordance with a second communication protocol, the method comprising the steps of:

(i) selecting a further device from a plurality of further devices in accordance with predetermined criteria,

(ii) retrieving a network address accessible to the selected further device for assigning to a network device in one of the networks, the retrieved network address being compatible with the communication protocol of the other network;

(iii) assigning the retrieved network address as an alias to the said network device;

(iv) applying said assigned alias to one or more communication sent from said network device; and

(v) sending the or each communication.

19. A method according to claim 18, further including monitoring device characteristics of the or each further devices, and comparing monitored device characteristics with the said predetermined criteria so as to select a said further device.

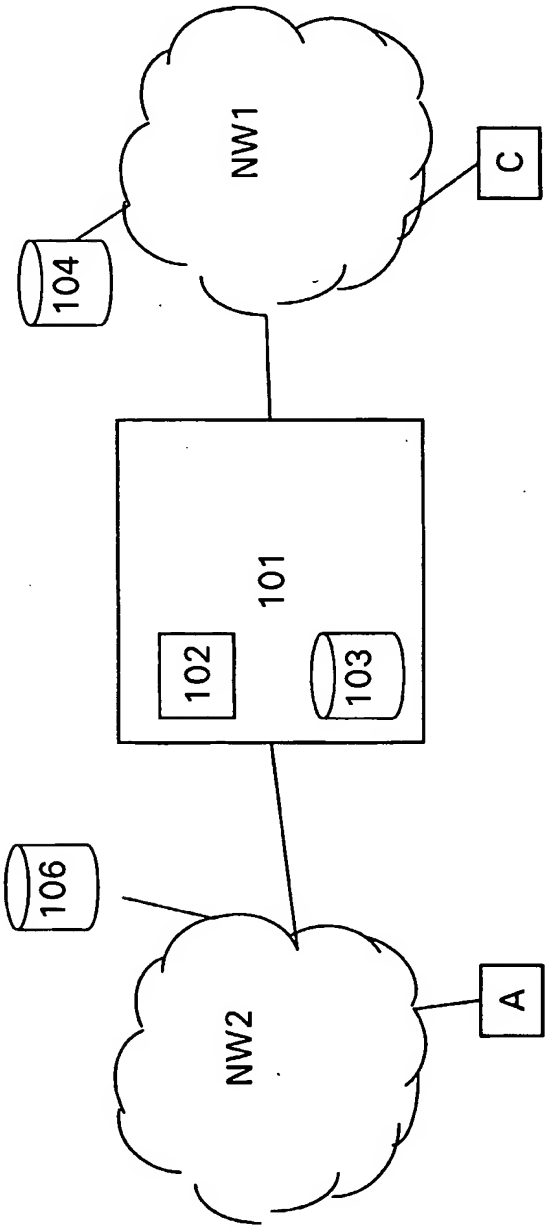


Fig 1

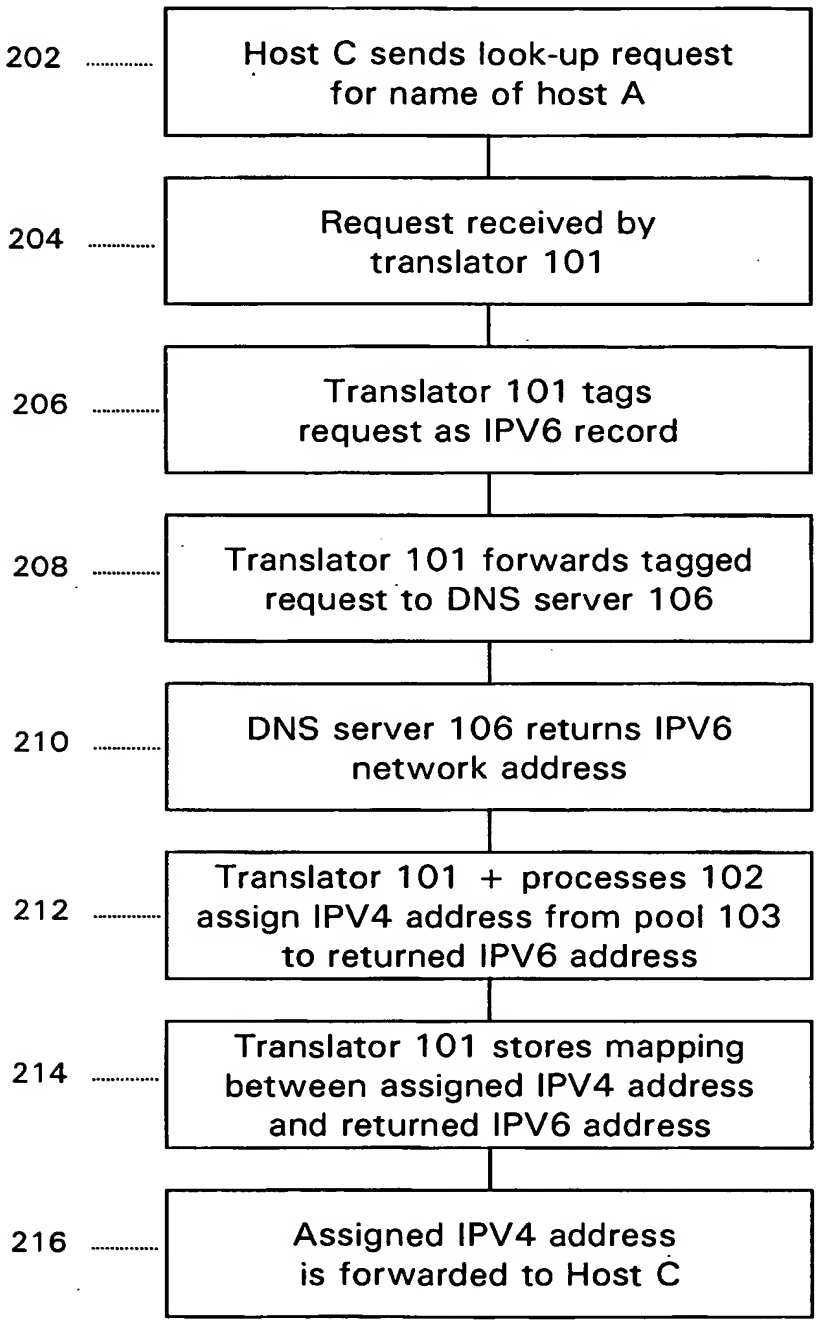

202
Host C sends look-up request for name of host A
204
Request received by translator 101
206
Translator 101 tags request as IPV6 record
208
Translator 101 forwards tagged request to DNS server 106
210
DNS server 106 returns IPV6 network address
212
Translator 101 + processes 102 assign IPV4 address from pool 103 to returned IPV6 address
214
Translator 101 stores mapping between assigned IPV4 address and returned IPV6 address
216
Assigned IPV4 address is forwarded to Host C

Fig 2

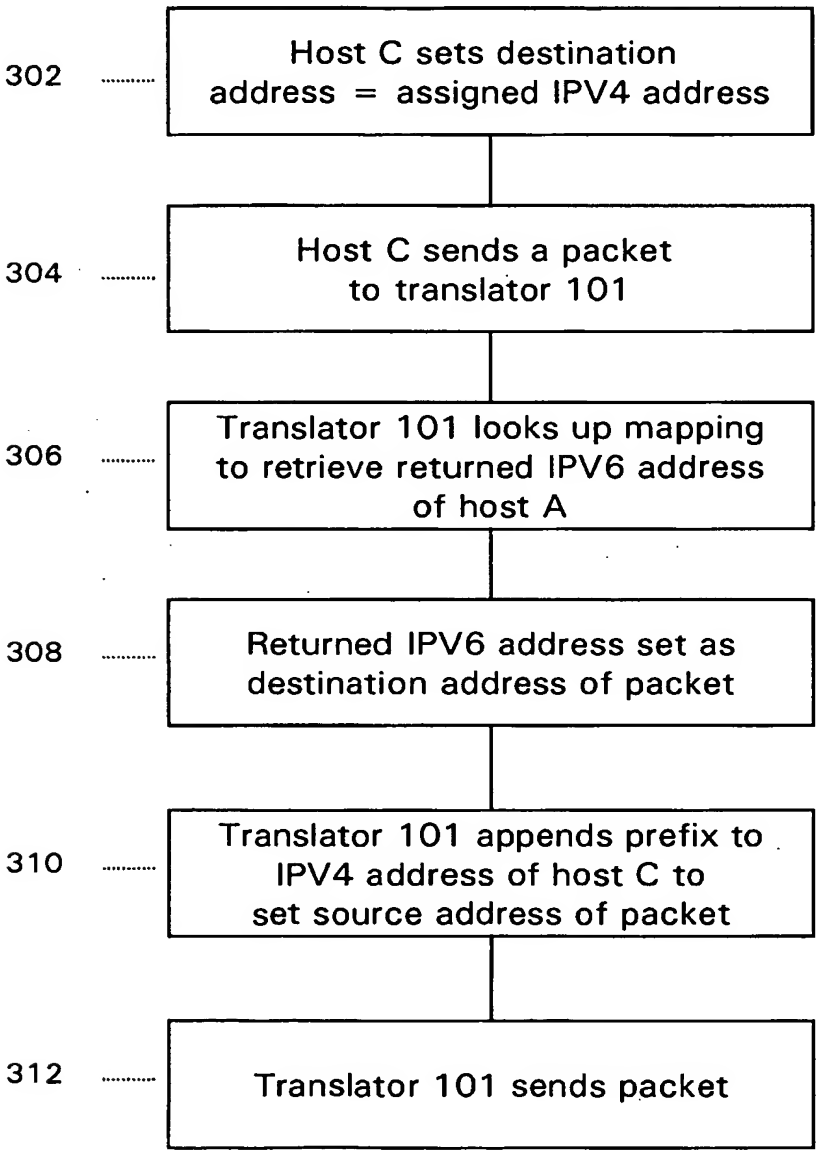

302
Host C sets destination address = assigned IPV4 address
304
Host C sends a packet to translator 101
306
Translator 101 looks up mapping to retrieve returned IPV6 address of host A
308
Returned IPV6 address set as destination address of packet
310
Translator 101 appends prefix to IPV4 address of host C to set source address of packet
312
Translator 101 sends packet

Fig 3

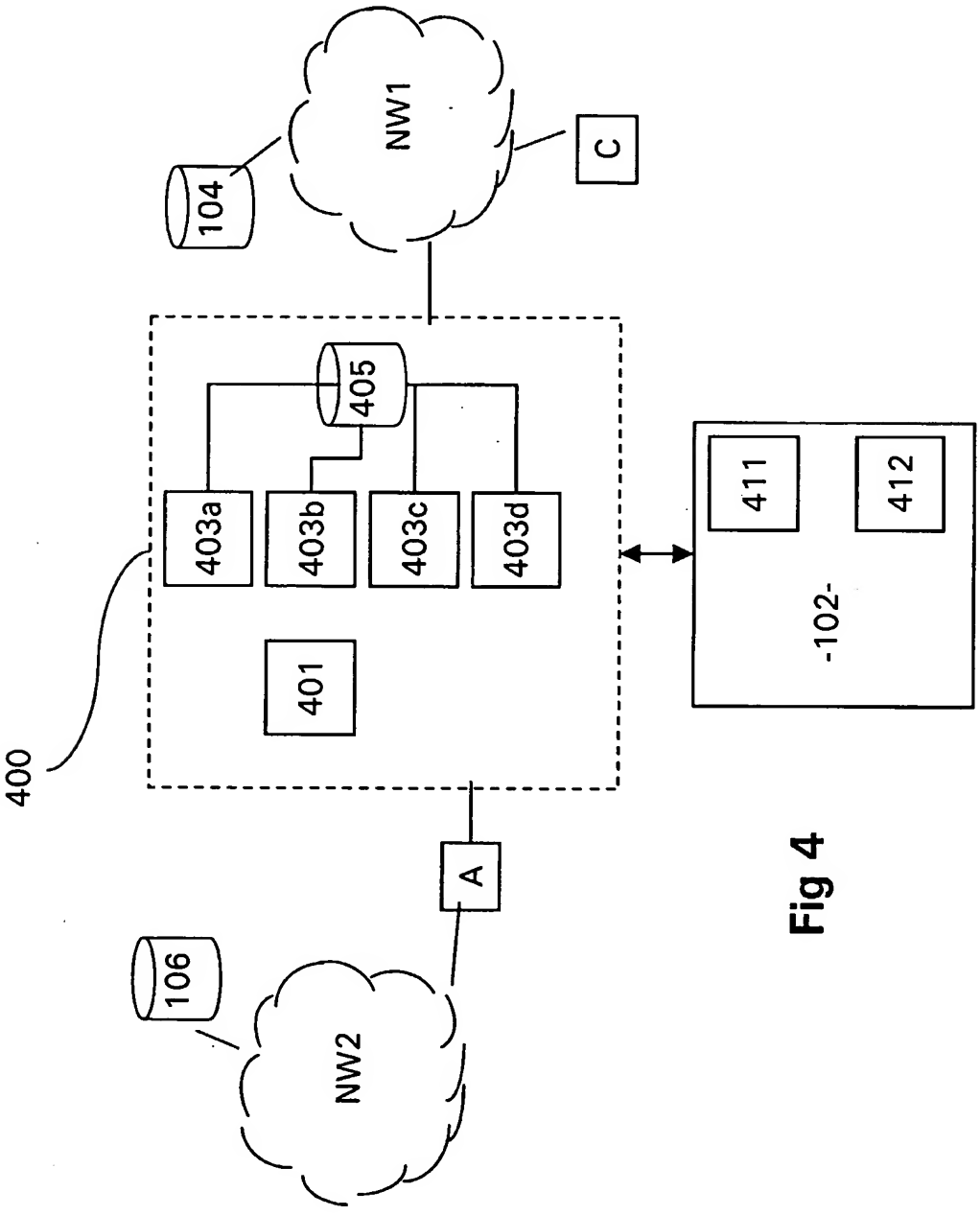
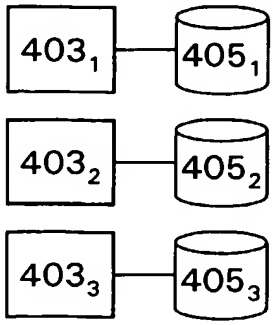


Fig 4



One – One
(1:1)

Fig 5a

2:1

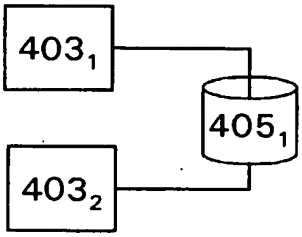


Fig 5b

3:1

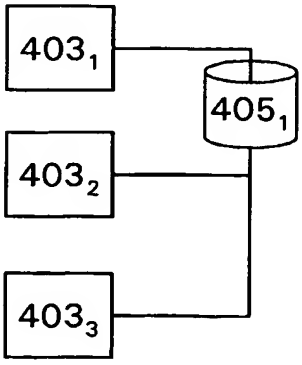


Fig 5c

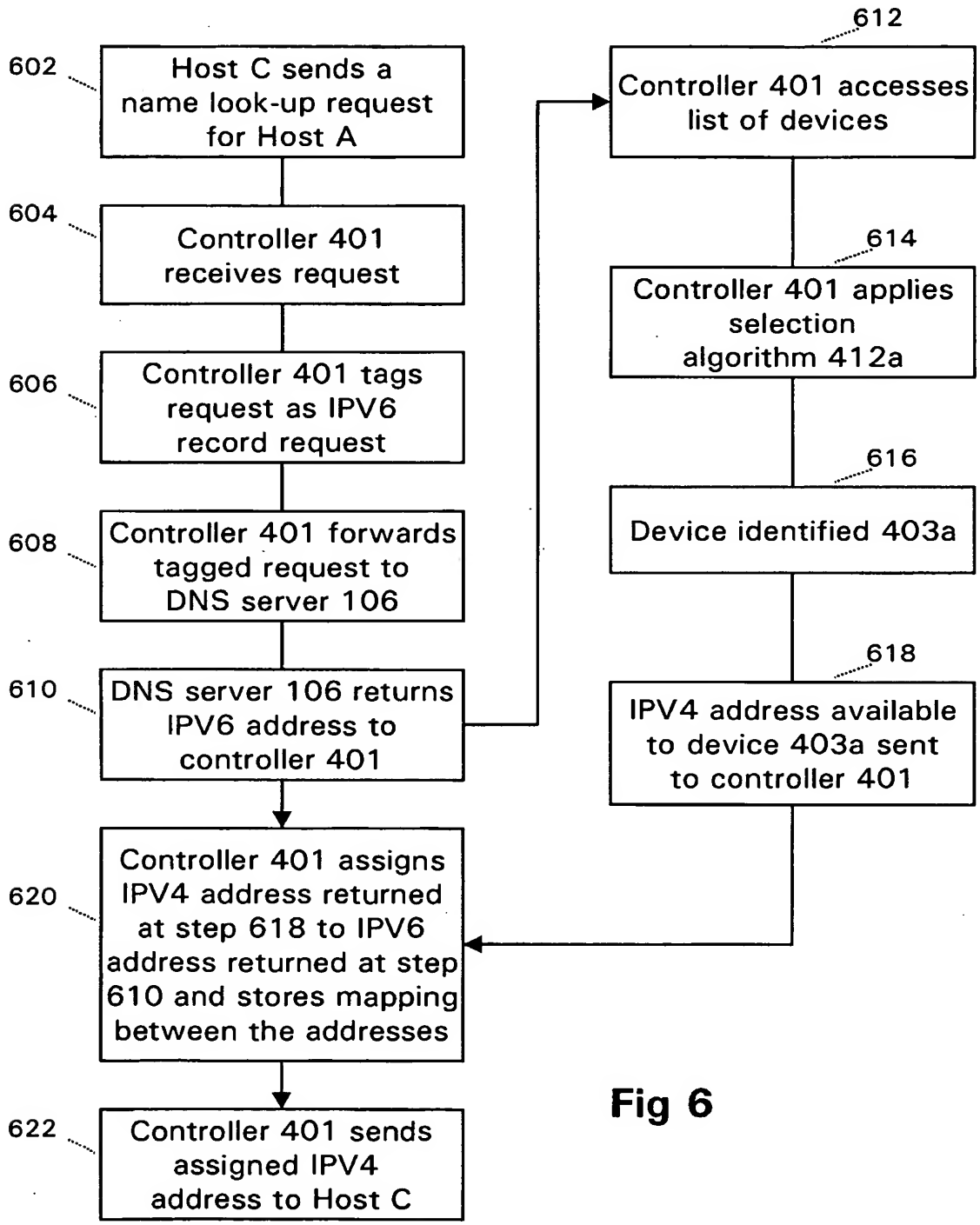

602
Host C sends a name look-up request for Host A
604
Controller 401 receives request
606
Controller 401 tags request as IPV6 record request
608
Controller 401 forwards tagged request to DNS server 106
610
DNS server 106 returns IPV6 address to controller 401
620
Controller 401 assigns IPV4 address returned at step 618 to IPV6 address returned at step 610 and stores mapping between the addresses
622
Controller 401 sends assigned IPV4 address to Host C
612
Controller 401 accesses list of devices
614
Controller 401 applies selection algorithm 412a
616
Device identified 403a
618
IPV4 address available to device 403a sent to controller 401

Fig 6

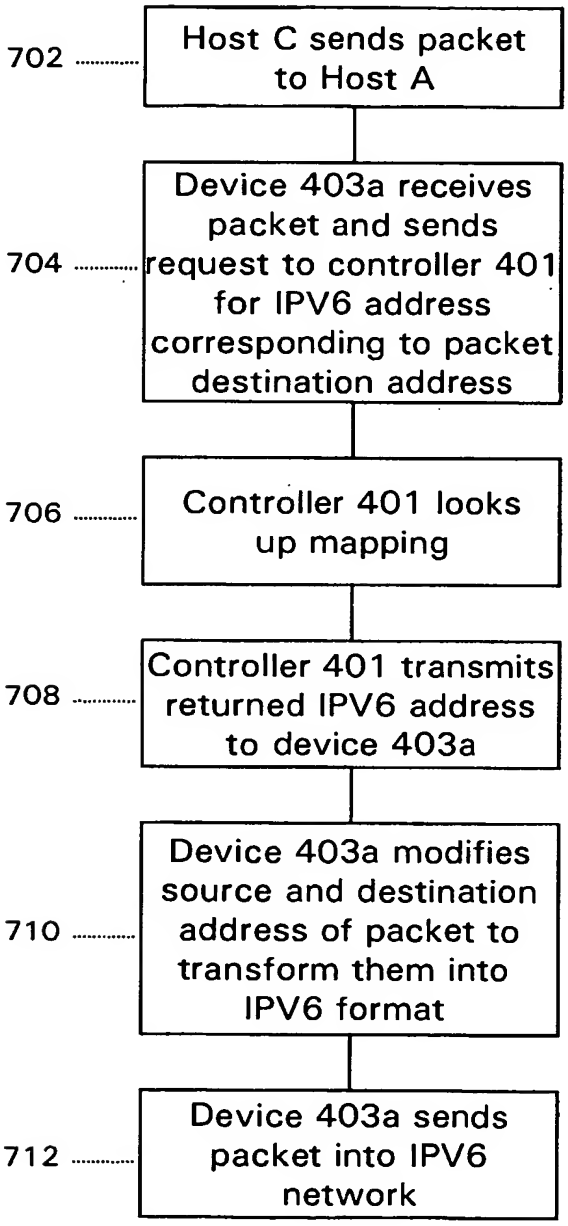

702
Host C sends packet to Host A
704
Device 403a receives packet and sends request to controller 401 for IPV6 address corresponding to packet destination address
706
Controller 401 looks up mapping
708
Controller 401 transmits returned IPV6 address to device 403a
710
Device 403a modifies source and destination address of packet to transform them into IPV6 format
712
Device 403a sends packet into IPV6 network

Fig 7

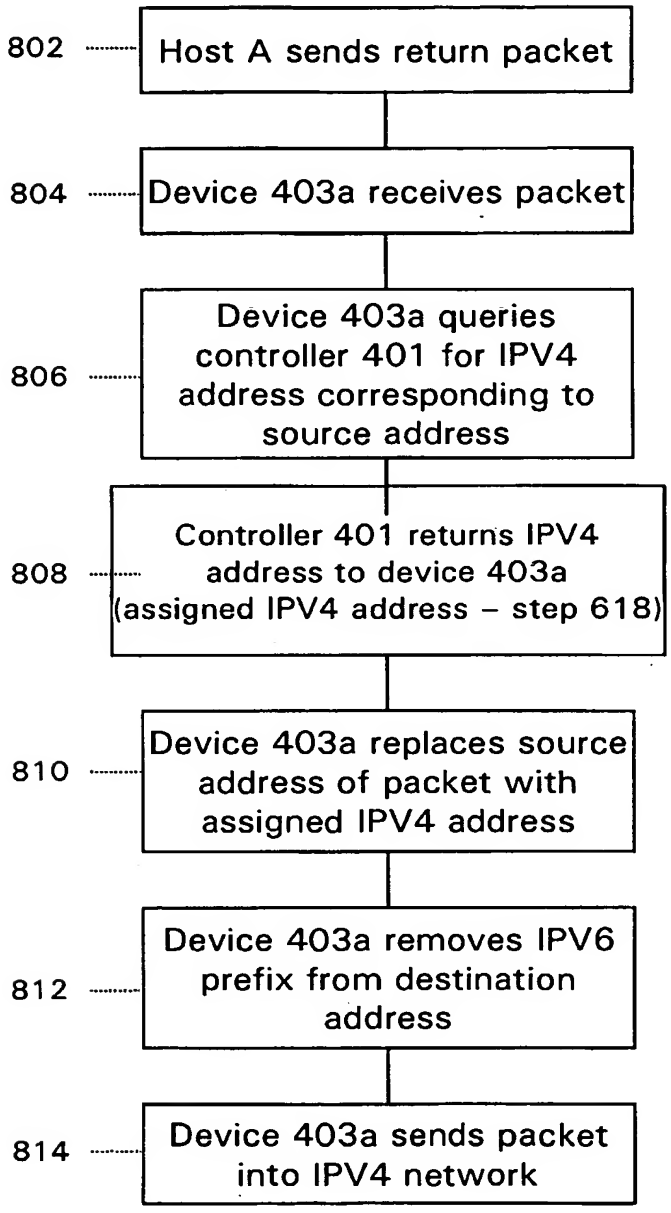

802
Host A sends return packet
804
Device 403a receives packet
806
Device 403a queries controller 401 for IPV4 address corresponding to source address
808
Controller 401 returns IPV4 address to device 403a (assigned IPV4 address – step 618)
810
Device 403a replaces source address of packet with assigned IPV4 address
812
Device 403a removes IPV6 prefix from destination address
814
Device 403a sends packet into IPV4 network

Fig 8

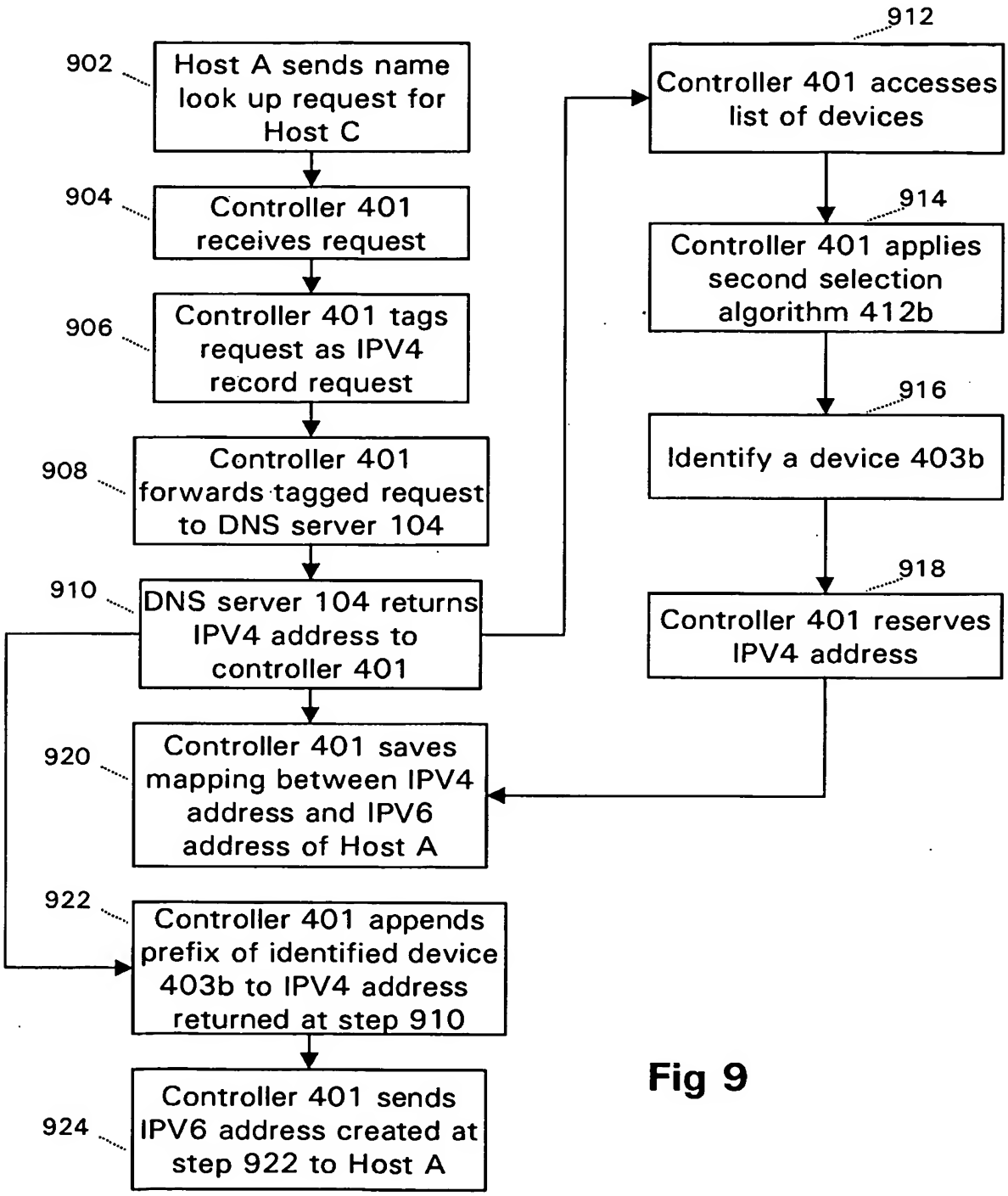

902
Host A sends name look up request for Host C
904
Controller 401 receives request
906
Controller 401 tags request as IPV4 record request
908
Controller 401 forwards tagged request to DNS server 104
910
DNS server 104 returns IPV4 address to controller 401
912
Controller 401 accesses list of devices
914
Controller 401 applies second selection algorithm 412b
916
Identify a device 403b
918
Controller 401 reserves IPV4 address
920
Controller 401 saves mapping between IPV4 address and IPV6 address of Host A
922
Controller 401 appends prefix of identified device 403b to IPV4 address returned at step 910
924
Controller 401 sends IPV6 address created at step 922 to Host A

Fig 9

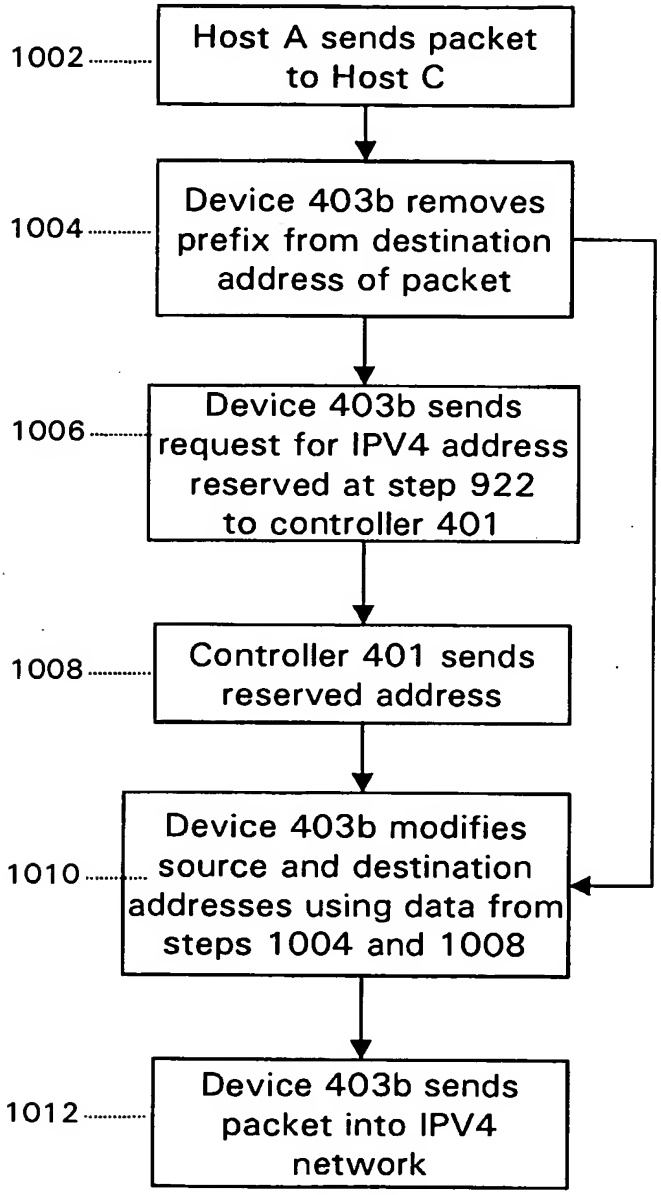

1002
Host A sends packet to Host C
1004
Device 403b removes prefix from destination address of packet
1006
Device 403b sends request for IPV4 address reserved at step 922 to controller 401
1008
Controller 401 sends reserved address
1010
Device 403b modifies source and destination addresses using data from steps 1004 and 1008
1012
Device 403b sends packet into IPV4 network

Fig 10

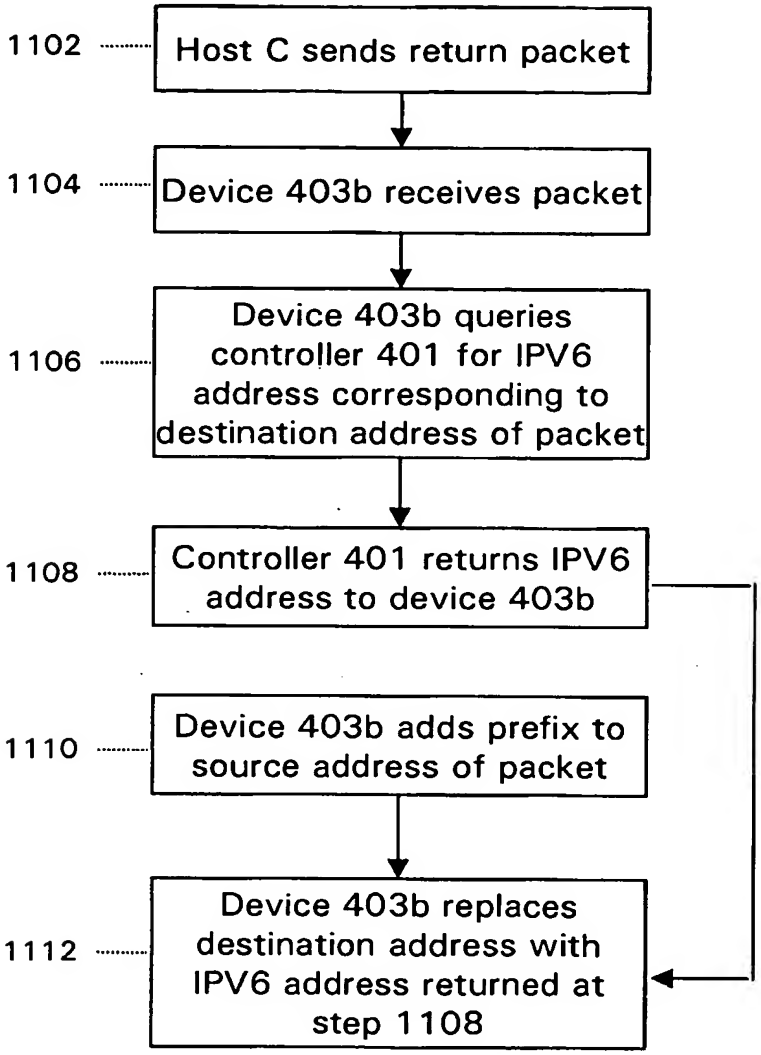

1102
Host C sends return packet
1104
Device 403b receives packet
1106
Device 403b queries controller 401 for IPV6 address corresponding to destination address of packet
1108
Controller 401 returns IPV6 address to device 403b
1110
Device 403b adds prefix to source address of packet
1112
Device 403b replaces destination address with IPV6 address returned at step 1108

Fig 11

INTERNATIONAL SEARCH REPORT

Int tional Application No
PCT/GB 02/00970

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L29/12 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	TSIRTSIS G ET AL: "RFC2766: Network Address Translation - Protocol Translation (NAT-PT)" REQUEST FOR COMMENT, February 2000 (2000-02), XP002167711 cited in the application abstract paragraph '001.! - paragraph '004.! paragraph '007.! - paragraph '07.1! ---	1-19
Y	WO 01 06734 A (3COM CORP) 25 January 2001 (2001-01-25) abstract page 1, line 1 -page 5, line 12 page 6, line 19-30 page 9, line 4-20 page 10, line 2 -page 13, line 32 figure 2 -----	1-19

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
18 July 2002	26/07/2002

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Lievens, K

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No
PCT/GB 02/00970

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 0106734 A	25-01-2001	EP 1201068 A2	02-05-2002
		WO 0106734 A2	25-01-2001